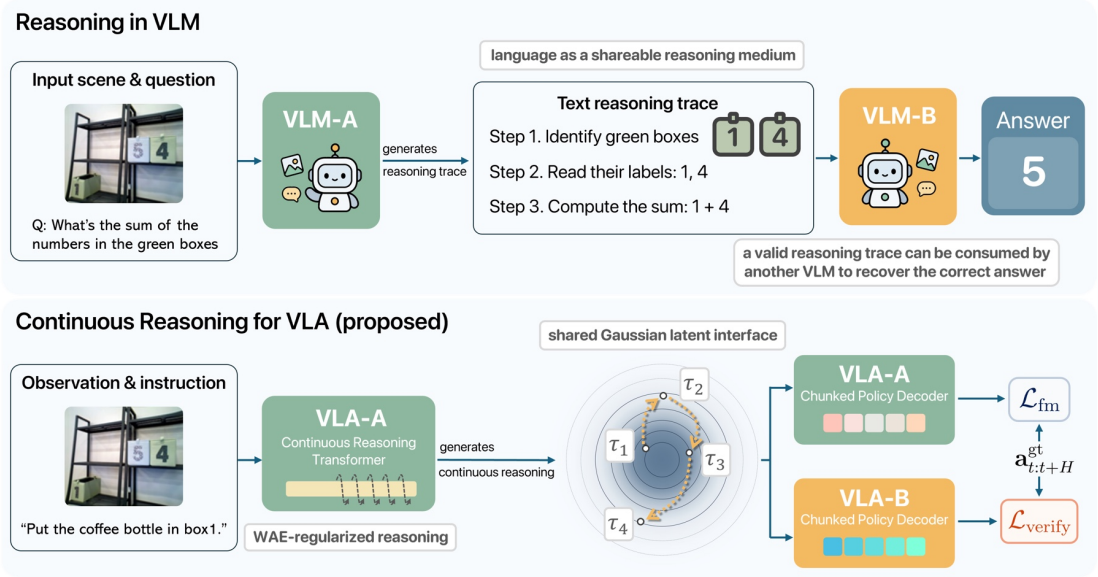
TECHNICAL REPORT

Continuous Reasoning for Vision-Language-Action

Yueh-Hua (Kris) Wu* **Tatsuya Matsushima** **Kei Ota**
AIRoA

Abstract

Natural language is a powerful reasoning medium for language and vision-language models, but it is mismatched to the granularity of continuous control. Text and explicit subgoals operate at task-level granularity, whereas vision-language-action (VLA) policies must choose actions at a much finer temporal scale; a single reasoning step can therefore span many action chunks while remaining only weakly coupled to the action needed now. This suggests a different question for VLA: what should play the role of language? We argue that a useful VLA reasoning medium must be shareable across model instances, verifiable through downstream action improvement, and aligned with temporally extended control structure. Based on this view, we propose *Continuous Reasoning for Vision-Language-Action*. Our model first predicts continuous reasoning in the form of a structured set of continuous thoughts, then reuses them as shared context for chunk-structured action generation. We instantiate continuous reasoning as a shared Gaussian latent interface and train it with a self-verification objective in which an exponential-moving-average teacher must successfully consume the student's reasoning when predicting target actions. Empirically, Continuous Reasoning improves robustness on LIBERO-PRO and performs strongly on both HSR and TX-G2 (an AgiBot G2-compatible variant), raising mean real-robot subtask success over $\pi_{0.5}$ by 40.4% on TX-G2 and 26.3% on HSR. These results suggest that reasoning in VLA is less about producing extra tokens and more about learning a shareable and verifiable internal language for action.



Top: in LLM/VLMs, language makes reasoning shareable and verifiable across model instances. Bottom: Continuous Reasoning seeks the analogous property for VLA through a shared Gaussian thought interface that another VLA instance must consume to predict better actions.

Keywords: Vision-language-action models, Reasoning

*Corresponding author: kris.wu@airoa.org.

1. Introduction

Vision-language-action (VLA) models have rapidly expanded the scope of robot policy learning by combining large-scale visual and language pretraining with action generation for manipulation and control. Recent systems have shown that internet-scale or multi-robot pretraining can substantially improve generalization, instruction following, and data efficiency in embodied decision making (Brohan et al., 2022, 2023; Driess et al., 2023; Octo Model Team et al., 2024; Kim et al., 2024; Black et al., 2025). At the same time, a parallel line of work has argued that successful embodied agents should not rely only on direct observation-to-action mapping, but should also perform some form of intermediate reasoning, planning, or decomposition before acting (Ahn et al., 2022; Huang et al., 2022; Liang et al., 2022; Huang et al., 2023). As VLA policies become stronger and more general, the central question is no longer whether reasoning matters, but what form that reasoning should take for continuous control.

Natural language offers a useful source of inspiration for this question. In language and vision-language models, reasoning is effective not only because language is interpretable, but because it provides a shared medium: a reasoning trace can be read, reused, and built upon by another person or another model. Good intermediate reasoning is therefore not merely internally helpful; it is reusable across instances and externally verifiable. This suggests a more precise question for embodied control: *what should play the role of language for VLA models?* If VLA systems are to reason before acting, they may also need a medium that preserves these functional advantages of language while remaining appropriate for the much finer granularity of continuous control.

Recent reasoning-oriented VLA methods implicitly propose several candidates for this medium, including textual or multimodal chain-of-thought traces (Zhao et al., 2025; Shou et al., 2026; Zhong et al., 2026; Wen et al., 2025; Yin et al., 2025), latent planning and compact reasoning states (Huang et al., 2025, 2026; Bai et al., 2026; Liu et al., 2026; Lee et al., 2025), action-level reasoning abstractions (Zhong et al., 2026), and interactive reasoning with external spatial guidance (Ling et al., 2026). Each captures part of what makes language effective, but they typically emphasize explicit traces, latent planning efficiency, or external guidance rather than a reasoning interface that is simultaneously shareable, verifiable, and aligned with control granularity. Textual traces are naturally reusable and shareable, yet they remain mismatched to the temporal scale of control. A reasoning step such as "first complete subtask 1, then subtask 2" may be meaningful at the task level, but each such step typically unfolds over many action chunks before the goal is reached. The same reasoning unit must therefore support many successive action outputs, making its contribution to the action needed at the current control instant weak and indirect. Language is also a poor carrier for the precise geometric and dynamical structure required by action prediction. Visual subgoals and explicit foresight can expose future structure, but they often specify where the policy should eventually go rather than which action chunk should be produced now, while also incurring significant prediction cost. Latent reasoning moves closer to our setting by internalizing intermediate computation in continuous space. But improved action prediction alone does not certify good reasoning. This concern already appears in language-model reasoning, where recent work shows that outcome-based or RL-trained reasoning traces can improve final-answer accuracy without reliably yielding reasoning that is faithful, verifiable, or causally important (Lanham et al., 2023; Mohammadi et al., 2025; Yu et al., 2026). In such cases, the added reasoning channel can help optimize behavior on seen problems without producing a process that is robust enough to generalize. The VLA setting inherits the same difficulty, but without a naturally shared medium like text; and text itself is mismatched to the granularity of continuous control. The right "language" for VLA must therefore do more than appear as an intermediate variable: it must match the structure of control.

In this paper, we therefore argue that the "language" of VLA should be modeled as a *continuous internal interface* with three properties. First, it should be *reusable*: if a reasoning trace is genuinely good, another model instance should be able to benefit from it rather than only the model that produced it. Second, it should be *shareable*: the representation should live in a common latent space that can be transmitted and consumed across model instances, rather than existing only as an opaque byproduct of one forward pass. Third, it should be *abstraction-aligned*: the representation should match the temporal granularity at which actions are organized, above low-level motor fluctuations but below free-form semantic language, so that it remains close enough to chunked action generation to guide control rather than merely describe it. Under this view, the goal of reasoning is not to generate extra text, but to form reusable symbols for future action generation. This also motivates our use of chunk-level action prediction as a natural unit for high-level control reasoning.

Based on this view, we propose *Continuous Reasoning for Vision-Language-Action*. Our model first predicts continuous reasoning as a structured set of continuous thoughts, then reuses those thoughts as a shared reasoning context for chunk-structured action generation. To reduce arbitrariness in the thought space, we regularize the thoughts with a Wasserstein autoencoder (WAE) into a shared latent geometry (Tolstikhin et al., 2017). To test whether the learned thoughts are genuinely reusable, we introduce a self-verification objective in which an exponential-moving-average (EMA) teacher must successfully consume the student's thoughts in order to predict target actions (Tarvainen and Valpola, 2017). This self-verification mechanism makes the reasoning interface operational rather than decorative: a thought is only valuable if another model instance can use it to act better. We further align reasoning with high-level control structure through chunked generation, horizon-aware verification, and training choices that require action prediction to depend on the learned thought interface.

In summary, this paper makes three contributions. First, we formulate *continuous reasoning* as a distinct interface for VLA, defined by reusability, shareability, and abstraction alignment. Second, we instantiate this view with a WAE-structured Gaussian latent interface, a self-verification objective, and chunk-aligned action generation. Third, we provide empirical evidence that these ingredients matter jointly, including mean real-robot subtask-success gains over $\pi_{0.5}$ of 40.4% on TX-G2, an AgiBot G2-compatible variant, and 26.3% on HSR, while sharpening what reasoning should mean for continuous robot control.

2. Related Work

Reasoning in VLA. Recent VLA research increasingly studies explicit intermediate reasoning rather than pure observation-to-action mapping. Existing approaches span language-like or multimodal chain-of-thought traces (Zhao et al., 2025; Shou et al., 2026; Zhong et al., 2026; Wen et al., 2025; Yin et al., 2025), latent planning and compact reasoning states (Huang et al., 2025, 2026; Bai et al., 2026; Liu et al., 2026; Lee et al., 2025), action-level reasoning abstractions (Zhong et al., 2026), and interactive reasoning with external spatial guidance (Ling et al., 2026). Related open-world and instruction-oriented systems also seek to preserve broader pretrained reasoning capabilities inside embodied models (Driess et al., 2023; Zhou et al., 2025; Yang et al., 2025; Qu et al., 2025). Our work is closest to latent and action-oriented reasoning approaches, but differs in treating the intermediate representation as a shared interface whose usefulness is tested by another model instance and whose structure is aligned with chunk-level control.

Backbones, action generation, and planning. General-purpose VLA models such as RT-1, RT-2, PaLM-E, Octo, OpenVLA, OpenVLA-OFT, π_0, and $\pi_{0.5}$establish the large-scale embodied policy-learning setting in which we operate (Brohan et al., 2022, 2023; Driess et al., 2023; Octo Model Team et al., 2024; Kim et al., 2024, 2025; Black et al., 2025; Intelligence et al., 2025). In parallel, robot policy learning has developed structured action-generation and planning paradigms, including chunked action prediction, behavior tokenization, diffusion-based control, language-guided decomposition, and latent subgoal structure (Zhao et al., 2023; Shafiullah et al., 2022; Lee et al., 2024; Chi et al., 2023; Hou et al., 2024; Haldar et al., 2024; Ahn et al., 2022; Huang et al., 2022; Liang et al., 2022; Huang et al., 2023, 2024; Liu et al., 2024; Chen et al., 2025; Lynch et al., 2019). Our focus is not primarily on scaling or decoding efficiency, but on what kind of reasoning interface should sit on top of a modern generative action policy.

Self-verification and latent structuring. Our training design is also related to teacher-student learning and structured latent modeling. Exponential-moving-average teachers are widely used to stabilize targets and encourage consistency across views or networks (Tarvainen and Valpola, 2017; Grill et al., 2020; Caron et al., 2021), while Wasserstein autoencoders impose a more organized latent geometry than unconstrained hidden states (Tolstikhin et al., 2017). Here their role is different: the EMA teacher tests whether predicted thoughts can be reused by another model instance, and WAE regularization supports a common latent space through which those thoughts can be shared.

3. Continuous Reasoning

We instantiate continuous reasoning as a shared internal interface between perception-language context and future action generation. Given an observation-instruction pair (o, x), the model first generates a small set of raw continuous thoughts over dedicated reasoning slots, maps them into shared Gaussian latent codes,

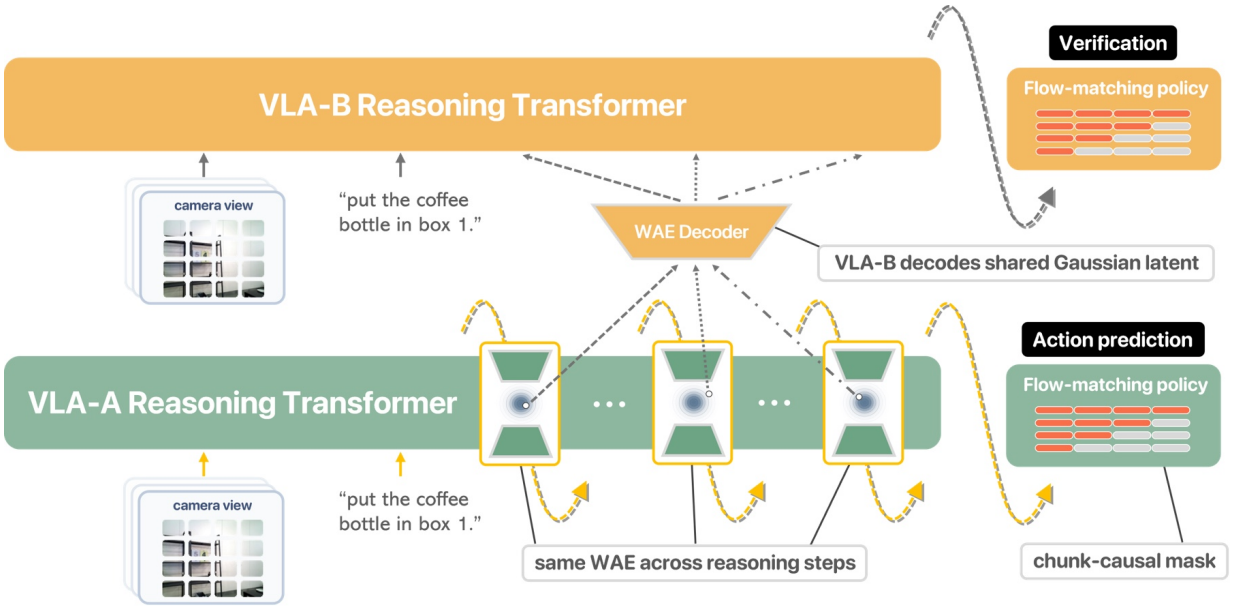


Figure 1 | Continuous Reasoning architecture. VLA-A generates continuous thoughts, maps them into shared Gaussian latent codes, and decodes them for chunk-causal action prediction. VLA-B decodes the same latent codes for verification; in our implementation, VLA-B is the EMA teacher.

and then decodes them into a reasoning interface that is reused for chunk-structured action generation under a flow-matching policy. This design separates global reasoning from local generation: reasoning is produced once per horizon before action decoding and then reused across the full rollout, rather than being consumed through a rigid one-thought-per-chunk mapping at generation time.

Figure 1 summarizes the full architecture. The central path generates raw thoughts slot by slot, regularizes them into shared Gaussian latent codes, and reuses the student-decoded reasoning interface for chunk-structured action generation. During training, an EMA teacher receives the same latent codes and decodes them with its own WAE decoder in order to verify that the shared interface is reusable across model instances.

Problem setup and backbone. We consider language-conditioned action prediction from observations o, instruction x, and future action sequence $a \in \mathbb{R}^{H\times d}$ using a chunked flow-matching backbone. We divide the action horizon into K temporally extended chunks, so that $H = KC$ for chunk size C. This chunked decomposition is important in our formulation: it defines the abstraction level at which reasoning should operate, above individual low-level action fluctuations but below free-form semantic language.

Instantiating continuous reasoning as structured thoughts. We represent continuous reasoning with N_τ raw thought vectors $\tau = [\tau_1, \dots, \tau_{N_\tau}]$, where each $\tau_i \in \mathbb{R}^D$ and N_τ is independent of the action-chunk count K. We refer to τ as the raw continuous thoughts. The thought slots are generated sequentially rather than jointly: each slot is computed from the current prefix state, its decoded interface is written back into the prefix, and that decoded state is committed to the KV cache so that later slots can attend to earlier ones. These thoughts are not decoded into text and are not assigned one-to-one to local action chunks. Instead, they provide a compact horizon-level reasoning scaffold that is reused globally during action generation.

Shareable latent geometry. To make this reasoning medium shareable across model instances, we regularize the raw thoughts with a WAE (Tolstikhin et al., 2017). An encoder maps the thoughts to shared latent codes via $z = q_\phi(\tau)$, and a decoder produces the student-side reasoning interface $\hat{\tau}^S = p_\psi(z)$. The EMA teacher receives the same shared latent codes z but decodes them with its own WAE decoder, with EMA decoder parameters $\bar{\psi}$, yielding a teacher-side interface $\hat{\tau}^{\text{ema}} = p_{\bar{\psi}}(z)$ before predicting actions. The WAE objective combines reconstruction and prior matching,

$$\mathcal{L}_{\text{wae}} = \lambda_{\text{rec}} \|\tau - \hat{\tau}^S\|_2^2 + \lambda_{\text{mmd}} \text{MMD}(z, \mathcal{N}(0, I)).$$

The purpose of this step is not to guarantee semantics by itself, but to impose a common latent geometry so that reasoning codes are less private and easier to transfer, decode, and reuse across model instances.

Flow-matching action prediction. Action prediction is trained with a flow-matching objective over noisy trajectories (Intelligence et al., 2025). Given ground-truth action sequence a, Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, and interpolation time $t \in (0, 1]$, we form the noisy action $x_t = (1-t)a + t\epsilon$ and target vector field $u_t = \epsilon - a$. The student action model predicts a vector field $v_\theta(o, x, x_t, t; \hat{\tau}^S)$ conditioned on the student-decoded reasoning interface, and is trained with

$$\mathcal{L}_{\text{fm}} = \mathbb{E}\left[\|v_\theta(o, x, x_t, t; \hat{\tau}^S) - u_t\|_2^2\right].$$

This decoder sits between fully bidirectional flow-matching action prediction and step-wise autoregressive decoding. Within each chunk, the flow-matching head predicts a temporally extended continuous action segment without left-to-right decoding over individual motor steps, following the broader line of chunked generative action policies (Zhao et al., 2023; Chi et al., 2023; Black et al., 2025; Kim et al., 2025). Across chunks, we use a block-causal attention mask: action tokens in the same chunk share one attention block, while later chunks can attend to earlier chunks. Thus temporal dependency is modeled at the level of high-level action chunks rather than microscopic control fluctuations, and the same decoded reasoning interface remains available throughout action generation.

Reusable reasoning through self-verification. To encourage the reasoning interface to be reusable rather than merely helpful to the producing model, we introduce a self-verification objective using an EMA teacher (Tarvainen and Valpola, 2017). Let $\bar{\theta}$ denote the EMA parameters, and let $\hat{\tau}^{\text{ema}}$ denote the teacher-decoded interface obtained from the student's shared latent codes. During training, the teacher must decode the same latent codes and use the resulting interface to predict the same target action field:

$$\mathcal{L}_{\text{verify}} = \mathbb{E}\left[\|v_{\bar{\theta}}(o, x, x_t, t; \hat{\tau}^{\text{ema}}) - u_t\|_2^2\right].$$

This objective operationalizes reusability at training time: another model instance must successfully decode the same shared latent codes and use the resulting interface when predicting target actions.

Abstraction alignment and interface dependence. The final requirement is that the reasoning medium capture the right level of abstraction for control. We enforce this in two ways. First, the chunk-level decoder places temporal dependency at the level of action chunks rather than individual motor steps: within each chunk, flow matching predicts a temporally extended action segment in parallel, while across chunks a block-causal mask allows later chunks to attend to earlier ones. Second, in the main configuration used throughout this paper, we do not use separate per-chunk conditioning paths that would let local action prediction bypass the shared reasoning interface, so action generation must consume the decoded reasoning prefix itself rather than a private shortcut. These choices keep continuous reasoning close to chunk-level control while discouraging it from collapsing into low-level motor perturbations.

Overall objective. The final training objective combines action prediction, latent structuring, and self-verification:

$$\mathcal{L} = \mathcal{L}_{\text{fm}} + \mathcal{L}_{\text{wae}} + \lambda_{\text{verify}}\mathcal{L}_{\text{verify}}.$$

Overall, the model first predicts raw continuous thoughts τ, maps them into shared Gaussian latent codes z, and then asks both the student policy and the EMA teacher to act from decoded interfaces derived from the same latent codes. The training objective therefore ties action prediction to a reasoning medium that is shared across model instances and aligned with chunk-level control, rather than to a producer-specific latent shortcut. We refer to this overall framework as *continuous reasoning* for VLA.

Table 1 | LIBERO-PRO perturbation breakdown across all four LIBERO suites.

Methods	libero_10				libero_goal				libero_object				libero_spatial			
	Obj.	Pos.	Sem.	Task	Obj.	Pos.	Sem.	Task	Obj.	Pos.	Sem.	Task	Obj.	Pos.	Sem.	Task
OpenVLA-OFT (Kim et al., 2025)	5.5	0.0	38.5	0.0	8.5	0.0	43.5	2.5	66.5	9.5	89.0	0.0	30.0	13.5	65.0	0.0
X-VLA (Zheng et al., 2025)	61.0	1.0	70.5	19.5	71.5	1.0	94.0	7.5	91.5	4.5	98.0	0.0	89.5	0.0	69.0	38.5
VLA-Adapter (Wang et al., 2025)	47.0	0.0	**91.0**	10.0	61.0	0.0	75.0	12.0	89.0	0.0	**99.0**	8.0	98.0	0.0	**98.0**	49.0
$\pi_{0.5}$ (Intelligence et al., 2025)	66.0	6.0	**91.0**	17.0	**90.0**	29.0	95.0	17.0	89.0	19.0	95.0	10.0	**99.0**	**53.0**	97.0	55.0
CR (Ours)	**66.5**	**15.5**	88.5	**28.5**	81.5	**36.0**	**96.0**	**33.0**	**93.0**	**54.5**	**99.0**	**27.0**	97.5	51.0	96.0	**60.0**

4. Experiments

We begin with LIBERO-PRO (Zhou et al., 2025), a robustness-oriented benchmark that is substantially more challenging than standard LIBERO evaluation because success depends not only on in-distribution task execution, but also on maintaining performance under controlled distribution shifts. Rather than measuring a single aggregate success rate, LIBERO-PRO exposes policies to multiple perturbation families across several LIBERO suites, making it possible to separate semantic robustness from spatial robustness and task-level generalization. This makes it a better testbed for our claim than standard LIBERO success rates, which are now often saturated by modern VLA systems.

Table 1 reports the main simulation comparison against strong VLA baselines. We report four official perturbation types used throughout this paper: *object*, which changes object appearance such as color, texture, or size; *position*, which changes object placement; *semantic*, which paraphrases the task instruction; and *task*, which evaluates transfer to different target task variants. Among these, *position* and *task* are the most revealing because they require the policy to re-anchor action under a changed layout or pursue a modified goal, rather than merely tolerate appearance shifts or instruction paraphrases.

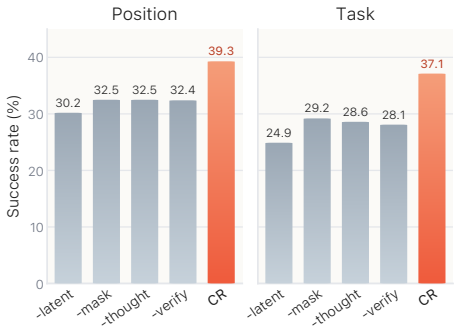


Figure 2 | LIBERO-PRO ablations. Labels indicate the removed component.

Continuous Reasoning yields the strongest overall robustness profile under the current LIBERO-PRO protocol, achieving the best suite mean on all four suites and improving the average suite mean from 58.0 with $\pi_{0.5}$ to 64.0. These gains come primarily from the two perturbations that most directly test action retargeting: averaged across suites, position success rises from 26.8 to 39.3 and task success from 24.8 to 37.1, while object and semantic performance remain competitive. In contrast, object and semantic perturbations can be spuriously forgiving under overfit action templates, because a policy may still obtain high scores by replaying nearly the same action sequence while largely ignoring the changed appearance or wording. The more revealing failure mode is therefore re-anchoring action under shifted spatial layouts or goal structure, and Continuous Reasoning improves exactly this regime.

Figure 4 provides a diagnostic view of the learned continuous reasoning interface beyond scalar success rates. We deliberately compare instruction pairs from the same initial scene and collect five rollouts for each instruction so that differences in the projected trajectories reflect task-dependent reasoning rather than scene variation. On the left, both tasks require reaching toward nearby objects and placing them into the same caddy compartment, so the early approach curves are similar without collapsing into an identical template. The trajectories become especially close near contact, where grasp geometry is similar for the cup and the book, and then separate again during placement because the larger book requires a different insertion strategy. On the right, the two tasks relocate different bowls to the same target plate. Here the pre-grasp portions differ substantially because the target bowls begin far apart, forcing different approach trajectories and reasoning before contact, while the post-grasp relocation phase exhibits a visibly related pattern across both instructions. This shared relocation structure appears specifically after the bowl has been grasped, suggesting that Continuous Reasoning is not merely encoding scene identity, but reorganizing around task phase and object-specific control de-

Figure 3 | TX-G2 and HSR.

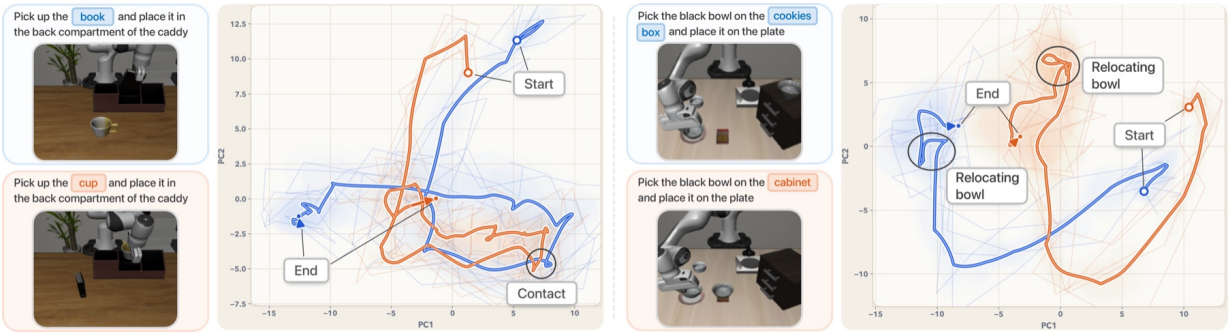


Figure 4 | PCA visualization of continuous reasoning trajectories on paired LIBERO-PRO scenes. Each instruction is evaluated with five rollouts from the same initial scene, so differences in the projected trajectories reflect instruction-dependent reasoning under matched scene context.

Table 2 | Real-robot subtask success on TX-G2 and HSR. Main-text results report mean subtask success with standard deviation; full E2E results are deferred to Appendix E.

Method	TX-G2				HSR			
	Cutl.	Bowl	Cloth.	Dish	Bott.-1	Bott.-2	Box	Mug
X-VLA	0.0 ±0.0	7.5 ±4.0	5.0 ±2.6	5.0 ±3.4	8.3 ±7.6	4.2 ±3.8	0.0 ±0.0	25.0 ±12.3
VLA-Adapter	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0
$\pi_{0.5}$	5.0 ±3.2	47.5 ±6.8	83.3 ±4.8	60.0 ±7.2	**83.3** ±9.6	45.8 ±8.5	41.7 ±14.0	66.7 ±9.6
CR (Ours)	**22.5** ±6.4	**70.0** ±7.2	**95.0** ±2.7	**87.5** ±4.5	**83.3** ±10.8	**83.3** ±6.8	**58.3** ±12.3	**75.0** ±10.2

mands. Additional paired-scene latent visualizations on LIBERO and TX-G2 are provided in Appendix C and Appendix E.

Ablation study. To isolate where these gains come from, we ablate Gaussian latent structuring, chunk-causal action masking, explicit continuous thoughts, and self-verification while keeping the same $\pi_{0.5}$ backbone and LIBERO-PRO evaluation protocol. Figure 2 focuses on the two perturbations that matter most for our claim, *position* and *task*, while the full table is provided in Appendix C. All four ablations reduce both *position* and *task* success, with the largest degradation appearing when Gaussian latent structuring is removed, whereas *object* and *semantic* remain comparatively stable. This again points to improved spatial re-anchoring and task-level adaptation rather than a generic regularization effect. We report CALVIN ABC→D in Appendix D as a complementary long-horizon benchmark, where Continuous Reasoning remains competitive, indicating that these LIBERO-PRO gains do not come at the expense of standard long-horizon execution.

Real-robot evaluation. We additionally evaluate Continuous Reasoning on two real-robot platforms, HSR and TX-G2 (an AgiBot G2-compatible variant). Figure 3 shows the two platforms used in our experiments. All reported real-robot baselines are fine-tuned for 150k steps. We exclude OpenVLA-OFT from this comparison because our RTX 5070 evaluation setup runs out of memory on both platforms under the deployed real-robot inference pipeline. In the main text, we report *Subtask* success, the mean success rate over subtasks within each task, and defer full E2E results, task descriptions, and protocol details to Appendix E.

TX-G2. TX-G2 is a bimanual platform with three cameras, and we query the deployed policy at 10 Hz. We evaluate each task with ten episodes, of which eight use in-distribution placements and two use substantially shifted out-of-distribution placements. For fairness, all methods are evaluated on the same ten initial states and object placements for each task. Objects may appear on either the left or right side of the workspace, so the policy must also decide which arm to use. We consider four long-horizon tasks: *Cutlery Transfer*, *Bowl Stacking*, *Clothes Sorting*, and *Dish Racking*.

Continuous Reasoning already shows a clear advantage over $\pi_{0.5}$ on TX-G2, improving subtask success on all four tasks. The gains are especially large on *Bowl Stacking* and *Dish Racking*, where multi-stage object handling and precise placement both matter. Appendix E shows that VLA-Adapter's 0.0 performance is more consistent with a weakly task-conditioned policy than with a major deployment bug.

Figure 5 provides a diagnostic view of continuous reasoning on the TX-G2 clothes-sorting subtask "pick

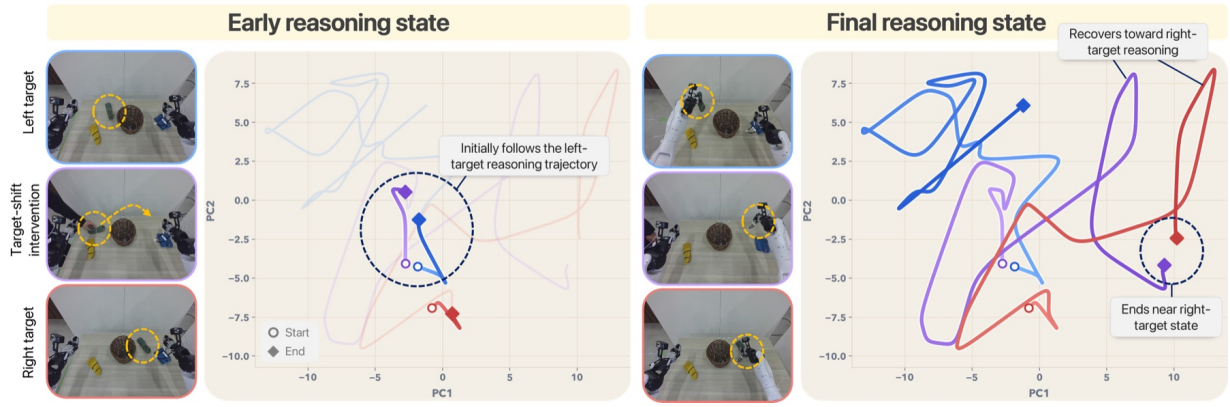


Figure 5 | PCA visualization of continuous reasoning on the TX-G2 subtask "pick up the green socks." Top: the target pair of green socks starts on the left. Bottom: it starts on the right. Middle: it starts on the left and is thrown to the right mid-episode by an external perturbation.

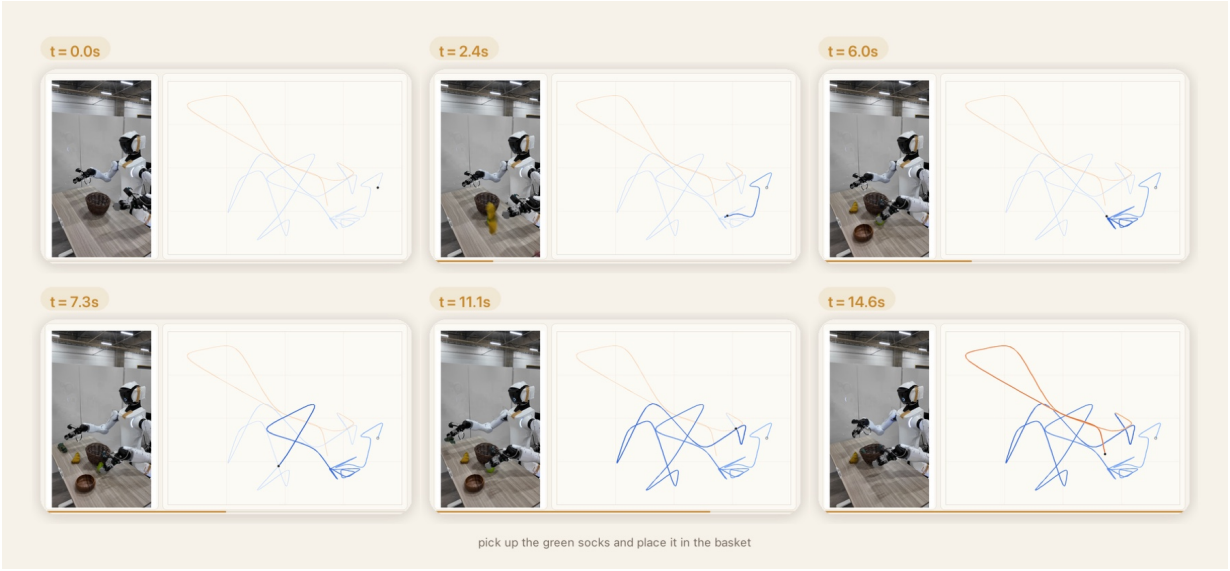


Figure 6 | TX-G2 latent visualization under dynamic object injection for the clothes-sorting task. Before the target green socks appear, the continuous reasoning latent remains clustered near the same state around t=6.0 despite incoming distractor objects. Once the green socks enter the scene, the latent sharply transitions into a different region and then proceeds through distinct pickup and placement phases. In the plot, blue markers denote the pickup stage and brown markers denote the placement stage, showing that the two phases require different latent reasoning states.

up the green socks." We visualize three matched scene variants: the target pair of green socks starts on the left (top), starts on the right (bottom), or starts on the left and is abruptly thrown to the right by an external perturbation during execution (middle). This kind of within-episode target displacement does not appear in the dataset. Nevertheless, the perturbed middle rollout initially follows the left-target reasoning pattern, but its final reasoning state moves toward the right-target configuration. This behavior is consistent with online re-anchoring of the reasoning interface after the target object has moved, rather than rigidly replaying the original plan.

Figure 6 probes a more adversarial TX-G2 setup in which the workspace initially contains no relevant object and a human successively throws different objects into view. Only green socks, yellow socks, a blue handkerchief, and the basket appear in the training data; at test time, we additionally inject distractors such as green fruit. The resulting latent trajectories reveal whether Continuous Reasoning stays inert under irrelevant events and reconfigures only when the true target becomes available.

HSR. HSR emphasizes long-horizon mobile manipulation with locomotion, and we query the deployed policy at 2 Hz. We use four tasks that combine navigation, manipulation, and constrained placement geometry: *Coffee Bottle* $\rightarrow$ *Box*, *Coffee Bottles* $\rightarrow$ *Table*, *Box Rearrangement*, and *Mug Rectangle*. Full task descriptions are deferred to Appendix E.

Continuous Reasoning also improves HSR subtask success on three of the four mobile-manipulation tasks, with the largest gain on *Coffee Bottles* $\rightarrow$ *Table*. This task is the most demanding in the suite because it requires repeated locomotion and manipulation over a long action chain: approach the shelf, pick a bottle, carry it to the table, return to the shelf, and repeat for the second bottle.

5. Conclusions

We presented Continuous Reasoning for Vision-Language-Action, a framework that treats reasoning in VLA not as an explicit language-like trace, but as a reusable continuous interface for future action generation. Our formulation learns structured continuous thoughts that are shared across model instances, organized in a common latent space, and reused as a reasoning context for chunk-structured action generation.

Our results support this view on both challenging generalization and real-robot evaluation. Continuous Reasoning improves robustness on LIBERO-PRO and performs strongly on both HSR and TX-G2, suggesting that reasoning for continuous control is better modeled as an internal language for action rather than as extra reasoning tokens. More broadly, we hope this work helps shift the discussion in VLA from whether models should reason before acting to what kind of reasoning interface they should learn.

Acknowledgments

This paper is based on results obtained from a project, JPNP25015, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We thank Ryosuke Takanami and Haru Kondoh for their assistance in the real-robot experiments.

References

A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2022. URL https://arxiv.org/abs/2212.06817.

A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. URL https://arxiv.org/abs/2307.15818.

D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. Palm-e: An embodied multimodal language model, 2023. URL https://arxiv.org/abs/2303.03378.

Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi,

Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

K. Black, N. Brown, D. Driess, A. Esmail, M. R. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, L. Smith, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can, not as i say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022. URL https://arxiv.org/abs/2207.05608.

J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control, 2022. URL https://arxiv.org/abs/2209.07753.

W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models, 2023. URL https://arxiv.org/abs/2307.05973.

Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, A. Handa, M.-Y. Liu, D. Xiang, G. Wetzstein, and T.-Y. Lin. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. 2025. doi: 10.48550/ARXIV.2503.22020. URL https://arxiv.org/abs/2503.22020.

Q. Shou, F. Zhu, S. Chen, P. Yan, Z. Yan, Y. Miao, X. Pang, Z. Hong, R. Shi, H. Huang, J. Zhang, and S. Guo. Halo: A unified vision-language-action model for embodied multimodal chain-of-thought reasoning, 2026. URL https://arxiv.org/abs/2602.21157.

Z. Zhong, J. Li, J. He, H. Yan, X. Gong, G. Zhao, Y. Cai, J. Gao, X. Yan, B. Liu, Y. Chen, L. Yang, and H. Li. Dualcot-vla: Visual-linguistic chain of thought via parallel reasoning for vision-language-action models, 2026. URL https://arxiv.org/abs/2603.22280.

J. Wen, M. Zhu, J. Liu, Z. Liu, Y. Yang, L. Zhang, S. Zhang, Y. Zhu, and Y. Xu. dvla: Diffusion vision-language-action model with multimodal chain-of-thought, 2025. URL https://arxiv.org/abs/2509.25681.

C. Yin, Y. Lin, W. Xu, S. Tam, X. Zeng, Z. Liu, and Z. Yin. Deepthinkvla: Enhancing reasoning capability of vision-language-action models, 2025. URL https://arxiv.org/abs/2511.15669.

C.-P. Huang, Y.-H. Wu, M.-H. Chen, Y.-C. F. Wang, and F.-E. Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning, 2025. URL https://arxiv.org/abs/2507.16815.

C.-P. Huang, Y. Man, Z. Yu, M.-H. Chen, J. Kautz, Y.-C. F. Wang, and F.-E. Yang. Fast-thinkact: Efficient vision-language-action reasoning via verbalizable latent planning, 2026. URL https://arxiv.org/abs/2601.09708.

S. Bai, J. Lyu, W. Zhou, Z. Li, D. Wang, L. Xing, X. Zhao, P. Wang, Z. Wang, C. Chi, B. Chen, and S. Zhang. Latent reasoning vla: Latent thinking and prediction for vision-language-action models, 2026. URL https://arxiv.org/abs/2602.01166.

Z. Liu, J. Liu, H. Chen, J. Yu, Z. Guo, C. Hou, C. Gu, X. Mi, R. Zhang, K. Wu, Z. Che, J. Tang, P.-A. Heng, and S. Zhang. Last$_0$: Latent spatio-temporal chain-of-thought for robotic vision-language-action model, 2026. URL https://arxiv.org/abs/2601.05248.

J. Lee, J. Duan, H. Fang, Y. Deng, S. Liu, B. Li, B. Fang, J. Zhang, Y. R. Wang, S. Lee, W. Han, W. Pumacay, A. Wu, R. Hendrix, K. Farley, E. VanderBilt, A. Farhadi, D. Fox, and R. Krishna. Molmoact: Action reasoning models that can reason in space, 2025. URL https://arxiv.org/abs/2508.07917.

L. Zhong, Y. Liu, Y. Wei, Z. Xiong, M. Yao, S. Liu, and G. Ren. Acot-vla: Action chain-of-thought for vision-language-action models, 2026. URL https://arxiv.org/abs/2601.11404.

Y. Ling, Q. Lian, J. Li, Q. Jiang, T. Zhang, X. Jiang, C. Liu, J. Liu, and L. Zhang. Guide, think, act: Interactive embodied reasoning in vision-language-action models, 2026. URL https://arxiv.org/abs/2605.13632.

T. Lanham, A. Chen, A. Radhakrishnan, B. Steiner, C. Denison, D. Hernandez, D. Li, E. Durmus, E. Hubinger, J. Kernion, K. Lukošiūtė, K. Nguyen, N. Cheng, N. Joseph, N. Schiefer, O. Rausch, R. Larson, S. McCandlish, S. Kundu, S. Kadavath, S. Yang, T. Henighan, T. Maxwell, T. Telleen-Lawton, T. Hume, Z. Hatfield-Dodds, J. Kaplan, J. Brauner, S. R. Bowman, and E. Perez. Measuring faithfulness in chain-of-thought reasoning, 2023. URL https://arxiv.org/abs/2307.13702.

H. Mohammadi, T. Kozak, and A. Giachanou. Evaluating GRPO and DPO for faithful chain-of-thought reasoning in LLMs, 2025. URL https://arxiv.org/abs/2512.22631.

Q. Yu, A. Tartaglini, P. Hase, C. Guestrin, and C. Potts. Outcome rewards do not guarantee verifiable or causally important reasoning, 2026. URL https://arxiv.org/abs/2604.22074.

I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders, 2017. URL https://arxiv.org/abs/1711.01558.

A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, 2017. URL https://arxiv.org/abs/1703.01780.

Z. Zhou, Y. Zhu, J. Wen, C. Shen, and Y. Xu. Chatvla-2: Vision-language-action model with open-world embodied reasoning from pretrained knowledge, 2025. URL https://arxiv.org/abs/2505.21906.

S. Yang, H. Li, B. Wang, Y. Chen, Y. Tian, T. Wang, H. Wang, F. Zhao, Y. Liao, and J. Pang. Instructvla: Vision-language-action instruction tuning from understanding to manipulation, 2025. URL https://arxiv.org/abs/2507.17520.

D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang, and X. Li. Spatialvla: Exploring spatial representations for visual-language-action model, 2025. URL https://arxiv.org/abs/2501.15830.

M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL https://arxiv.org/abs/2502.19645.

P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone, 2022. URL https://arxiv.org/abs/2206.11251.

S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafiullah, and L. Pinto. Behavior generation with latent actions. 2024. doi: 10.48550/ARXIV.2403.03181. URL https://arxiv.org/abs/2403.03181.

C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2023. URL https://arxiv.org/abs/2303.04137.

Z. Hou, T. Zhang, Y. Xiong, H. Pu, C. Zhao, R. Tong, Y. Qiao, J. Dai, and Y. Chen. Diffusion transformer policy, 2024. URL https://arxiv.org/abs/2410.15959.

S. Haldar, Z. Peng, and L. Pinto. Baku: An efficient transformer for multi-task policy learning, 2024. URL https://arxiv.org/abs/2406.07539.

Z. Huang, Y. Lin, F. Yang, and D. Berenson. Subgoal diffuser: Coarse-to-fine subgoal generation to guide model predictive control for robot manipulation, 2024. URL https://arxiv.org/abs/2403.13085.

L. Liu, W. Wang, Y. Han, Z. Xie, P. Yi, J. Li, Y. Qin, and W. Lian. Foam: Foresight-augmented multi-task imitation policy for robotic manipulation, 2024. URL https://arxiv.org/abs/2409.19528.

H. Chen, J. Guo, B. Wang, T. Zhang, X. Huang, B. Zheng, Y. Hou, C. Tie, J. Deng, and L. Shao. Goal-vla: Image-generative vlms as object-centric world models empowering zero-shot robot manipulation, 2025. URL https://arxiv.org/abs/2506.23919.

C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play, 2019. URL https://arxiv.org/abs/1903.01973.

J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020. URL https://arxiv.org/abs/2006.07733.

M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers, 2021. URL https://arxiv.org/abs/2104.14294.

X. Zhou, Y. Xu, G. Tie, Y. Chen, G. Zhang, D. Chu, P. Zhou, and L. Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization, 2025. URL https://arxiv.org/abs/2510.03827.

J. Zheng, J. Li, Z. Wang, D. Liu, X. Kang, Y. Feng, Y. Zheng, J. Zou, Y. Chen, J. Zeng, Y.-Q. Zhang, J. Pang, J. Liu, T. Wang, and X. Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025. URL https://arxiv.org/abs/2510.10274.

Y. Wang, P. Ding, L. Li, C. Cui, Z. Ge, X. Tong, W. Song, H. Zhao, W. Zhao, P. Hou, S. Huang, Y. Tang, W. Wang, R. Zhang, J. Liu, and D. Wang. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model, 2025. URL https://arxiv.org/abs/2509.09372.

A. Experimental Configuration

Unless otherwise noted, all experiments in this paper use the same default H16 training configuration summarized in Table A1. This includes the main LIBERO-PRO comparison, the LIBERO-PRO ablations, the CALVIN results, and the real-robot experiments. The benchmark-specific sections below therefore focus on evaluation protocol details rather than on different model settings.

Table A1 | Default H16 training configuration used across all experiments in this paper unless otherwise noted.

Symbol	Value	Setting
H	16	action horizon
C	4	action chunk length
K	4	number of chunks ($H = KC$)
N_τ	2	maximum thought slots
d_z	128	thought latent dimension
λ_{verify}	0.1	constant verification weight
γ_{EMA}	0.994	EMA decay
$\mathcal{S}_{\text{curr}}$	$[0, 2000]$	thought curriculum steps

Distributed WAE-MMD implementation. In multi-GPU, multi-node training, the WAE prior-matching loss should not be computed independently on local shards. Instead, we first aggregate the latent codes and Gaussian prior samples across all devices to form one global batch, and then compute a single IMQ-kernel MMD loss on that global batch. This detail matters because MMD is a batch-level discrepancy: if each device computes its own local MMD, the kernel only sees small per-device sub-batches and no longer matches the intended global latent distribution. We use this same distributed WAE computation across all experiments in this paper unless otherwise noted.

B. LIBERO-PRO Evaluation Details

Our LIBERO-PRO results follow the official static benchmark protocol released with LIBERO-PRO. We evaluate on four suites: `libero_spatial`, `libero_goal`, `libero_10`, and `libero_object`. Each suite contains 10 tasks. For every suite–perturbation pair, we run 20 trials per task, yielding 200 evaluation episodes for each cell reported in Table 1.

We report four official perturbation types: *object*, *position*, *semantic*, and *task*. Success rates are reported as percentages over the corresponding 200 episodes. The *Suite Mean* in Table 1 is the arithmetic mean of these four perturbation success rates within the same suite block.

This protocol is intentionally more diagnostic than a single in-distribution success rate. Because each perturbation family isolates a different source of shift, the full LIBERO-PRO table reveals whether a policy mainly benefits from appearance robustness, instruction robustness, spatial retargeting, or task-level generalization.

C. LIBERO-PRO Ablation Details

Table A2 reports the full LIBERO-PRO ablation breakdown averaged across all four suites. The variants correspond to removing one major component of Continuous Reasoning at a time while keeping the same $\pi_{0.5}$ backbone and training budget.

Table A2 | Full LIBERO-PRO ablation results averaged across the four suites. The full Continuous Reasoning model is lightly highlighted for readability.

Method	Object	Position	Semantic	Task	Overall
Full CR	84.6	**39.3**	94.9	**37.1**	**64.0**
w/o Gaussian latent space	86.0	30.2	95.5	24.9	59.2
w/o chunk-causal mask	**86.5**	32.5	96.0	29.2	61.1
w/o continuous thoughts	84.9	32.5	**96.5**	28.6	60.6
w/o self-verification	86.0	32.4	95.5	28.1	60.5

For clarity, we summarize the precise implementation of each ablation below.

Ablation definitions. **w/o self-verification** removes the verification loss L_{verify} and does not train against the EMA teacher. The model still predicts continuous thoughts and still uses latent structuring, but those thoughts are no longer required to improve another model instance.

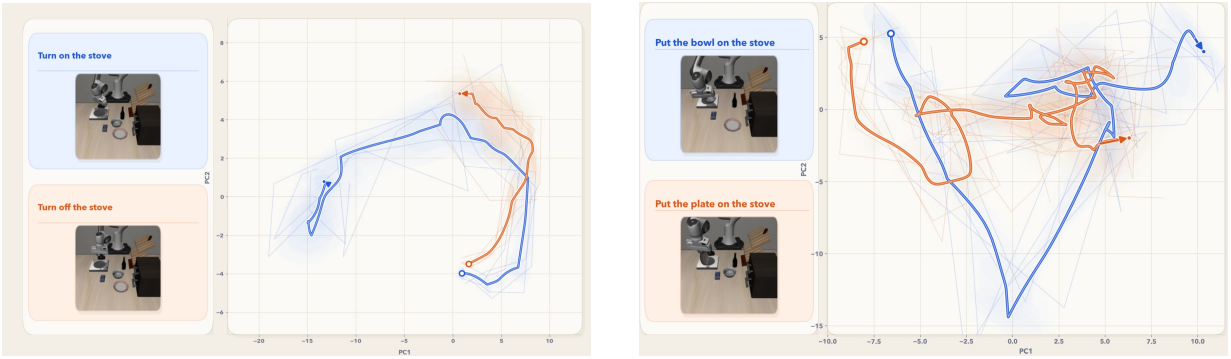
w/o Gaussian latent space removes the WAE objective and the Gaussian bottleneck. Instead of decoding thoughts from a structured latent code, we pass the model's original uncompressed continuous thought vectors directly to the action predictor and to the EMA teacher. This tests whether latent structuring itself matters beyond simply having a continuous intermediate representation.

w/o chunk-causal mask keeps continuous thoughts, latent structuring, and self-verification, but removes the block-causal dependency across action chunks. In this variant, the flow-matching decoder treats the full action horizon as a single block rather than imposing the AR-like temporal dependency used by the full method.

w/o continuous thoughts is the action-only variant. We remove the continuous reasoning branch entirely and keep only the chunk-structured flow-matching action decoder. Because no thought interface is produced, this variant also has no latent-structuring objective and no self-verification objective.

The same pattern as the main-text plot is visible here. Object and semantic perturbations remain relatively stable, but every ablation substantially reduces position and task success. This supports the claim that Continuous Reasoning primarily improves the components of control that require spatial retargeting and adaptation to changed task structure.

Additional LIBERO paired-scene latent diagnostics. We additionally visualize paired LIBERO scenes that share the same initial state but differ only in instruction, to further probe how the latent reasoning interface reorganizes under changed goals. These plots are diagnostic rather than mechanistic proofs, but they provide a more direct qualitative view of what the learned latent preserves and where it diverges.



Turn on vs. turn off the stove

Place different objects onto the stove

Figure A1 | Additional LIBERO paired-scene latent visualizations under matched initial states. Left: the two instructions initially produce closely aligned latent trajectories because both begin with the same approach-to-stove motion; the *turn off* rollout then terminates early because the stove already starts in the off state. Right: when placing different objects onto the stove, the latent state is similar near the initial and final phases, where the start and placement geometry are analogous, but diverges in the middle while the policy reaches to different object locations.

D. Additional Results on CALVIN

We report CALVIN ABC→D as a complementary long-horizon benchmark. While it is not the primary benchmark used to support our main empirical claims, it provides a useful check that Continuous Reasoning remains competitive on standard long-horizon manipulation evaluation. We report only average sequence length here.

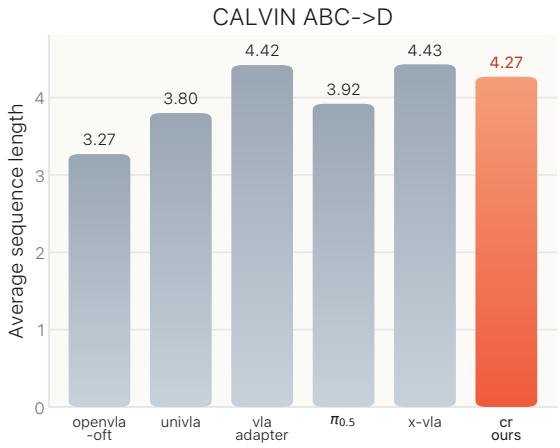


Figure A2 | CALVIN ABC→D average sequence length.

As shown in Figure A2, Continuous Reasoning remains close to the strongest baselines on CALVIN ABC→D, indicating that the robustness gains observed in the main text are not purchased by sacrificing standard long-horizon performance.

E. Real-Robot Evaluation Details

Shared evaluation setup. All reported real-robot baselines are fine-tuned for 150k steps. We exclude OpenVLA-OFT from both TX-G2 and HSR because our RTX 5070 evaluation setup runs out of memory under the deployed real-robot inference pipeline.

TX-G2 protocol. TX-G2 (an AgiBot G2-compatible variant) is a bimanual platform equipped with three cameras. Each task is evaluated with 10 episodes: 8 use in-distribution object placements that remain close to the training distribution without exactly duplicating dataset states, and 2 use substantially shifted out-of-distribution placements. For fairness, all compared methods are evaluated on the same 10 initial states and object placements for each task. Objects may appear on either the left or right side of the workspace, so the policy must also decide which arm to use.

TX-G2 dataset composition. The TX-G2 finetuning dataset contains 1198 source trajectories across four long-horizon task families. Table A3 summarizes the trajectory count for each family, and the short descriptions below give the canonical subtask sequence that appears most often in each case.

TX-G2 tasks. **Cutlery Transfer** requires moving a spoon and a fork from one bowl to another. The utensils are thin and difficult to grasp, and lifting the source bowl counts as failure.

Bowl Stacking presents three bowls with different colors and asks the robot to stack the instructed target bowls.

Clothes Sorting places two pairs of socks together with a handkerchief on the table and requires the instructed items to be placed into a basket.

Dish Racking uses two colored plates and requires placing the instructed plate onto a dish rack with limited clearance, making orientation correction particularly important.

Table A3 | Composition of the TX-G2 finetuning dataset.

Task family	Trajectories
Bowl Stacking	129
Clothes Sorting	514
Cutlery Transfer	206
Dish Racking	349

Canonical TX-G2 subtask sequences. **Bowl Stacking.** A typical sequence is: pick up the yellow bowl from the desk, stack the yellow bowl on the grey bowl, pick up the light-blue bowl from the desk, and stack the light-blue bowl on the yellow bowl.

Clothes Sorting. A typical sequence is: pick up the green socks from the desk, place the green socks into the basket, pick up the handkerchief from the desk, place the handkerchief into the basket, pick up the yellow socks from the desk, and place the yellow socks into the basket.

Cutlery Transfer. A typical sequence is: pick up the light-blue spoon from the grey bowl, place the spoon in the yellow bowl, pick up the pink fork from the grey bowl, and place the fork in the yellow bowl.

Dish Racking. A typical sequence is: pick up the yellow dish from the desk, place the yellow dish in the wooden dish rack, pick up the green dish from the desk, and place the green dish in the wooden dish rack.

Figure A3 | Representative TX-G2 trajectory snapshots for the four long-horizon tasks described above. Each panel shows a rollout of the corresponding task family used in our real-robot evaluation.

Additional TX-G2 closed-loop robustness under human intervention. We also evaluate a more challenging TX-G2 rollout in which the workspace initially starts empty and the policy must follow a strict ordered instruction sequence: pick up the green socks and place them into the basket, then the handkerchief, and finally the yellow socks. During execution, a nearby human continuously perturbs the scene by moving the basket and introducing additional objects. This setting is substantially harder than the standard clothes-sorting evaluation because the policy must both identify the correct target and preserve the required ordering under ongoing scene changes.

HSR tasks. HSR emphasizes long-horizon mobile manipulation with locomotion.

HSR dataset composition. The HSR finetuning dataset contains 1205 source trajectories across four long-horizon mobile-manipulation task families. Table A4 summarizes the trajectory count for each family, and the short descriptions below give the canonical subtask sequence that appears most often in each case.

Table A4 | Composition of the HSR finetuning dataset.

Task family	Trajectories
Mug Rectangle	399
Coffee Bottle → Box	295
Box Rearrangement	195
Coffee Bottles → Table	316

Coffee Bottle → Box requires HSR to move from a start position to a coffee bottle, grasp it, navigate to the area in front of the designated box, and insert the bottle into that box. This task is difficult because the clearance above the target box is small, so locomotion error makes precise placement substantially harder.

Coffee Bottles → Table requires moving two coffee bottles from a shelf to the table and placing them near the mugs. This task combines locomotion with repeated pick-and-place under clutter, and requires stable transport from an elevated shelf to a lower placement surface.

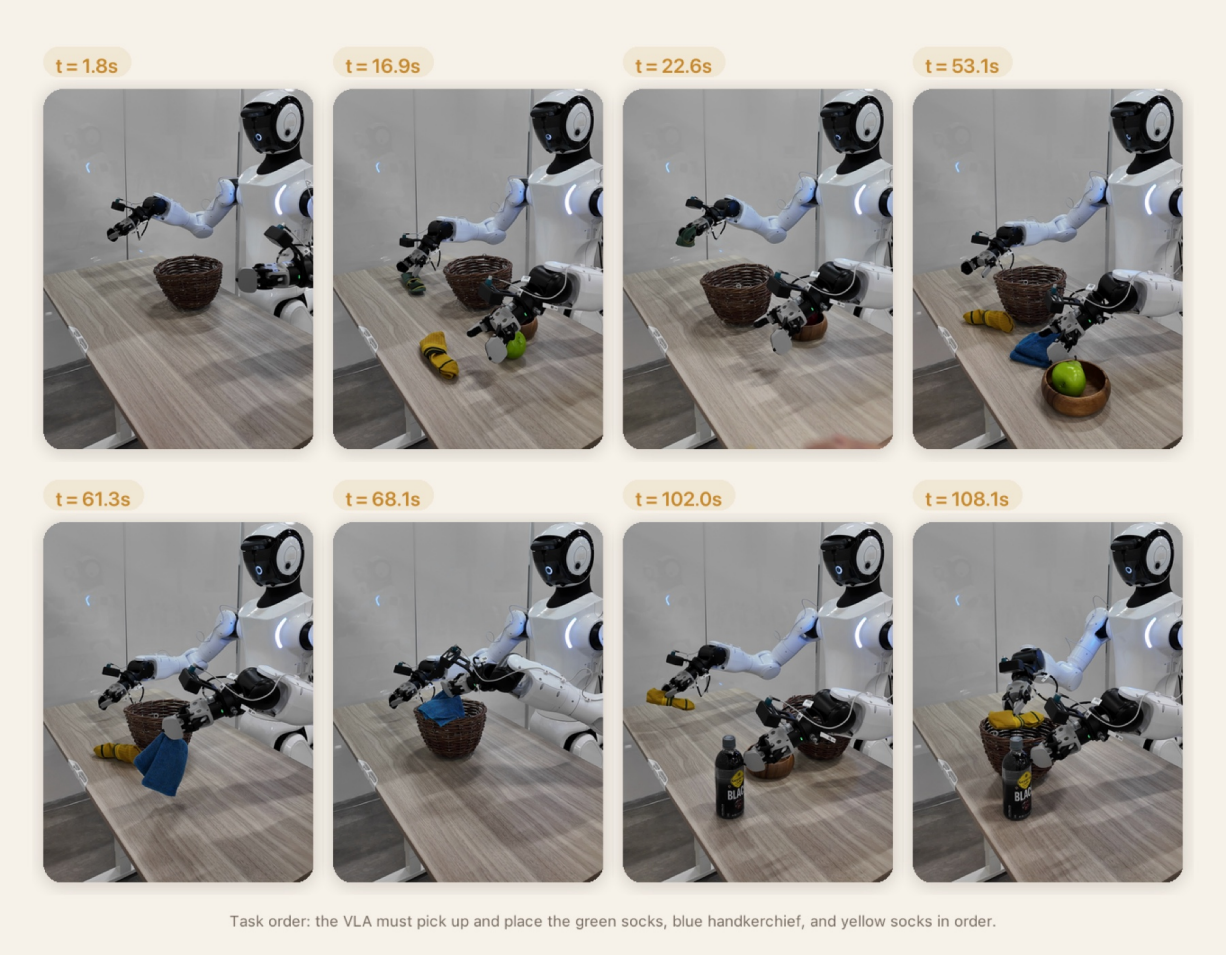


Figure A4 | TX-G2 closed-loop rollout under continuous human intervention. The table starts empty, and a nearby human repeatedly moves the basket and inserts additional objects while the policy must still follow the ordered sequence *green socks* → *handkerchief* → *yellow socks*. Despite these perturbations, Continuous Reasoning stably completes the required sequence and places the instructed items into the basket in order.

Box Rearrangement requires moving the box labeled 2 beside the box labeled 1. Its subtasks are moving to box 2, grasping box 2, navigating to the area in front of box 1, and placing box 2 beside box 1. The box is large and relatively heavy for the gripper, making stable transport and final placement both challenging.

Mug Rectangle requires moving mugs so that the final layout forms a rectangular arrangement. The policy must reason about which mug should move and where the missing rectangle corner lies before executing the relocation.

Canonical HSR subtask sequences. **Mug Rectangle.** A typical sequence is: pick up the mug that is not at a rectangle corner, and place it at the missing rectangle corner.

Coffee Bottle → Box. A typical sequence is: pick up the coffee bottle on the right, and place it into the box labeled 1.

Box Rearrangement. A typical sequence is: pick up the box labeled 2, and place it beside the box labeled 1.

Coffee Bottles → Table. A typical sequence is: pick up the right-hand bottle from the shelves, place it next to any mug on the table, pick up the left-hand bottle from the shelves, and place it next to a mug that does not already have a bottle.

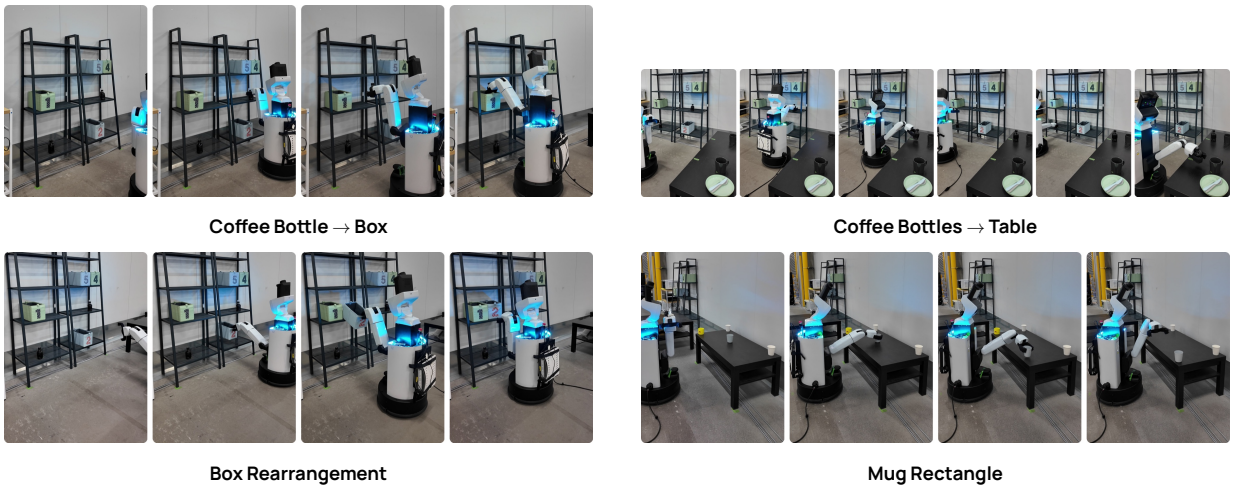


Figure A5 | Representative HSR trajectory snapshots for the four mobile-manipulation tasks described above. Each panel shows a rollout of the corresponding task family used in our real-robot evaluation.

Full real-robot tables. In the main-text real-robot table, TX-G2 shorthand headers *Cutl.*, *Bowl*, *Cloth.*, and *Dish* denote *Cutlery Transfer*, *Bowl Stacking*, *Clothes Sorting*, and *Dish Racking*, respectively. HSR shorthand headers *Bott.-1*, *Bott.-2*, *Box*, and *Mug* denote *Coffee Bottle → Box*, *Coffee Bottles → Table*, *Box Rearrangement*, and *Mug Rectangle*, respectively.

Table A5 reports the full TX-G2 metrics, including both mean subtask success and E2E completion. Table A6 provides the corresponding HSR table with the same metrics.

Table A5 | Full TX-G2 real-robot results on four long-horizon tasks. Subtask denotes mean subtask success with standard deviation, while E2E measures full-task completion when all subtasks succeed consecutively.

Method	Cutlery Transfer		Bowl Stacking		Clothes Sorting		Dish Racking	
	Subtask	E2E	Subtask	E2E	Subtask	E2E	Subtask	E2E
X-VLA	0.0 ±0.0	0.0	7.5 ±4.0	0.0	5.0 ±2.6	0.0	5.0 ±3.4	0.0
VLA-Adapter	0.0 ±0.0	0.0	0.0 ±0.0	0.0	0.0 ±0.0	0.0	0.0 ±0.0	0.0
$\pi_{0.5}$	5.0 ±3.2	0.0	47.5 ±6.8	0.0	83.3 ±4.8	50.0	60.0 ±7.2	20.0
CR (Ours)	**22.5** ±6.4	**0.0**	**70.0** ±7.2	**40.0**	**95.0** ±2.7	**80.0**	**87.5** ±4.5	**60.0**

Why does VLA-Adapter fail on TX-G2? We did not find evidence of a large inference-time integration error for VLA-Adapter. Its preprocessing path, proprioceptive normalization, and bundle-level input/output contract all passed our validation checks. However, in a 30-sample dataset-replay sanity test, the predicted actions were only marginally better than a simple mean-action baseline. This suggests that the adapted policy remained close to an average regressor rather than becoming a strongly task-conditioned controller, which is consistent with its 0% closed-loop success on TX-G2.

Table A6 | Full HSR real-robot results on four long-horizon mobile-manipulation tasks. Subtask denotes mean subtask success with standard deviation, while E2E measures full-task completion when all subtasks succeed consecutively.

Method	Bottle Box		Two Bottles		Box Rearr.		Mug Rect.	
	Subtask	E2E	Subtask	E2E	Subtask	E2E	Subtask	E2E
X-VLA	8.3 ±7.6	0.0	4.2 ±3.8	0.0	0.0 ±0.0	0.0	25.0 ±12.3	16.7
VLA-Adapter	0.0 ±0.0	0.0	0.0 ±0.0	0.0	0.0 ±0.0	0.0	0.0 ±0.0	0.0
$\pi_{0.5}$	**83.3** ±9.6	**66.7**	45.8 ±8.5	0.0	41.7 ±14.0	**16.7**	66.7 ±9.6	33.3
CR (Ours)	**83.3** ±10.8	**66.7**	**83.3** ±6.8	**66.7**	**58.3** ±12.3	**16.7**	**75.0** ±10.2	**50.0**